# Anexo Técnico: Metodología, Validación y Caso de Negocio

Este documento constituye el apéndice técnico del informe final. Detalla la arquitectura de datos, la validación estadística de hipótesis y la proyección financiera del proyecto SAREP.

## A. Estrategia de Datos y Preprocesamiento

Para abordar el problema de **"Arranque en Frío"** (*Cold Start*) derivado de la falta de históricos centralizados en la UNRC, se implementó una estrategia híbrida:

1. **Análisis Descriptivo (Contexto):** Se procesaron los datos longitudinales de los informes de *Numeralia UNRC (2021-2025)* para diagnosticar la brecha de eficiencia terminal (Sección 2.2.1 del informe principal).
2. **Modelado Predictivo (Solución):** Se aplicó una técnica de **Adaptación de Dominio** utilizando un *Dataset Proxy Estandarizado* (UCI Machine Learning Repository) para entrenar el motor de inferencia inicial.

### Pipeline de Ingeniería de Características (src/data/data_processing.py)

El pipeline de transformación garantiza la calidad de los datos para el algoritmo XGBoost:

- **Imputación de Nulos:** Estrategia estadística conservadora (Mediana para variables numéricas asimétricas, Moda para categóricas).
- **Codificación:**
    - *One-Hot Encoding:* Para variables nominales sin orden (ej. Intención de abandono (binarias), Carrera).
    - *Label Encoding:* Para variables ordinales (ej. Nivel de satisfacción vocacional).
- **Escalado:** *Min-Max Scaling* para normalizar rangos y mejorar la convergencia del modelo.
- **Manejo de Desbalance:** Se aplicó **SMOTE** (Synthetic Minority Over-sampling Technique) en el conjunto de entrenamiento para compensar el sesgo hacia la clase mayoritaria.

---

## B. Validación Estadística de Hipótesis (Inferencia)

Se realizó un análisis descriptivo básico con procesamiento en CSV de las encuestas y limpieza de registros inválidos. Posteriormente, se aplicaron pruebas no paramétricas para validar la relevancia estadística de los factores de riesgo, triangulando los hallazgos del modelo proxy con datos locales.

### B.1 Prueba de Independencia Chi-Cuadrado ($\chi^2$)

**Objetivo:** Evaluar si la deserción es estadísticamente dependiente de factores académicos y vocacionales.

$ \chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} $

**Resultados de las Hipótesis Evaluadas:**

| | Estadístico P- |
|---|---|

| Hipótesis | $\chi^2$ | valor | Interpretación Estadística |
|---|---|---|---|
| **H1: Rendimiento Académico (1er Semestre)** | 5.8101 | 0.0547 | **Asociación Marginal.** La tendencia es visible, aunque limitada por el tamaño muestral en subgrupos de reprobación severa. |
| **H3: Alineación Vocacional** | **6.7594** | **0.0341** | **Significativa ($p<0.05$).** Se rechaza la hipótesis nula; el desajuste vocacional es un predictor activo. |

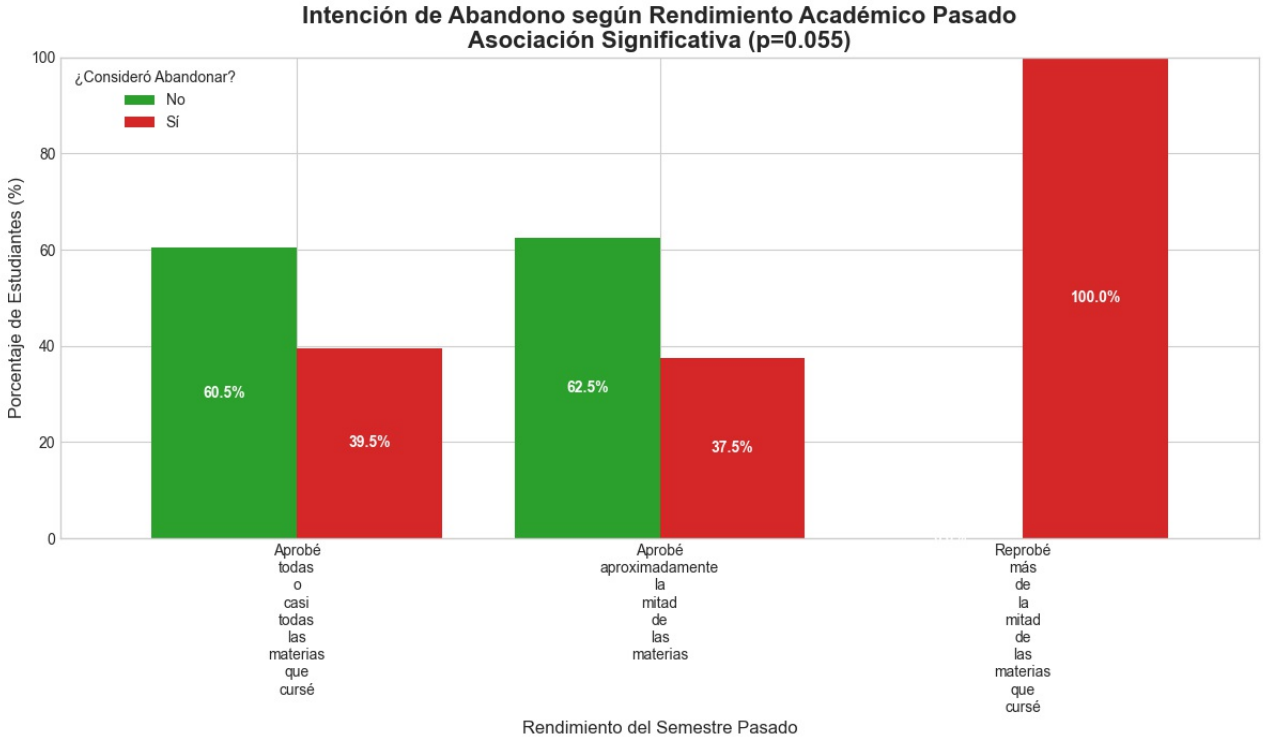


*Fig 1. Distribución porcentual de deserción según nivel de rendimiento.*

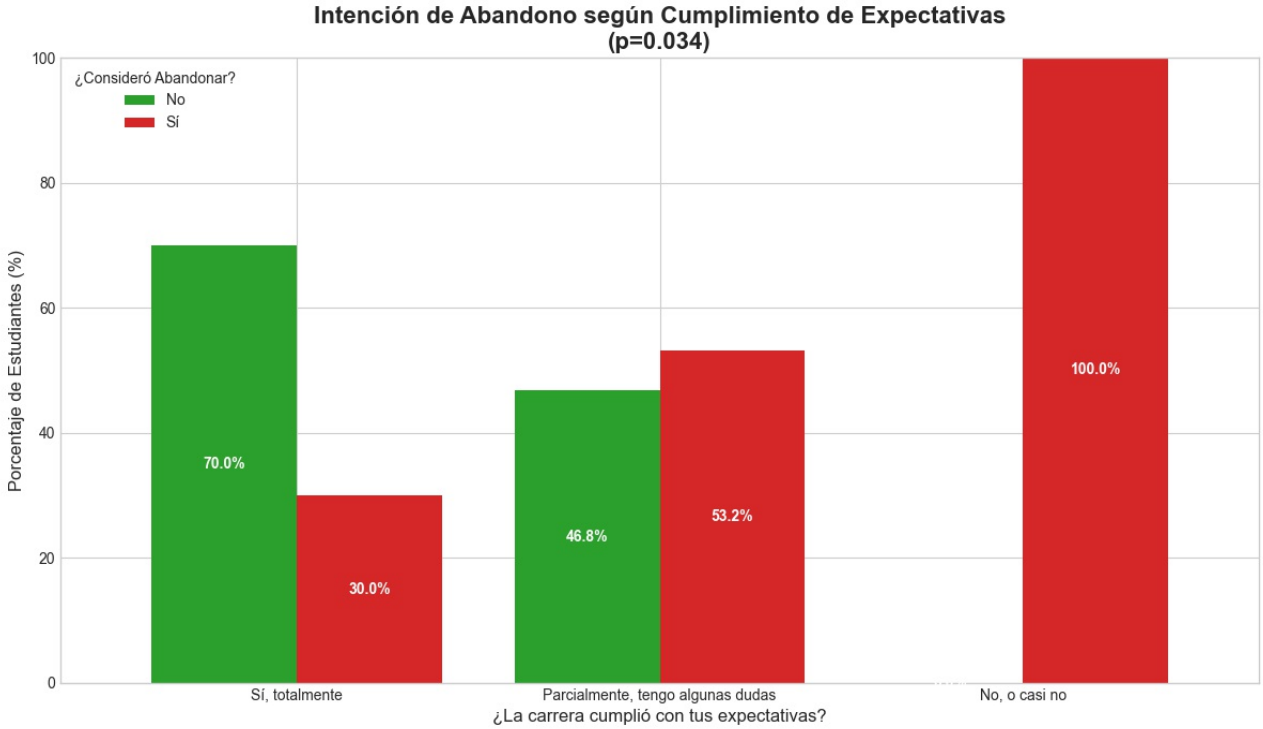


*Fig 2. Impacto de las expectativas vocacionales en la retención.*

### B.2 Prueba H de Kruskal-Wallis (Intensidad)

**Objetivo:** Analizar si el bajo rendimiento aumenta la *intensidad* de los pensamientos de abandono (escala ordinal 1-5), no solo su presencia.

$ H = (N-1) \frac{\sum_{i=1}^{g} n_i (\bar{r}*{i\cdot} - *bar{r})^2}{\sum*{i=1}^{g} \sum_{j=1}^{n_i} (r_{ij} - \bar{r})^2} $

**Resultados:**

- **Estadístico H:** 8.3955
- **P-valor: 0.0150** (Altamente Significativo)
- **Conclusión:** Existe una diferencia de varianza estocástica entre los grupos; a menor rendimiento, mayor intensidad en la ideación de abandono.

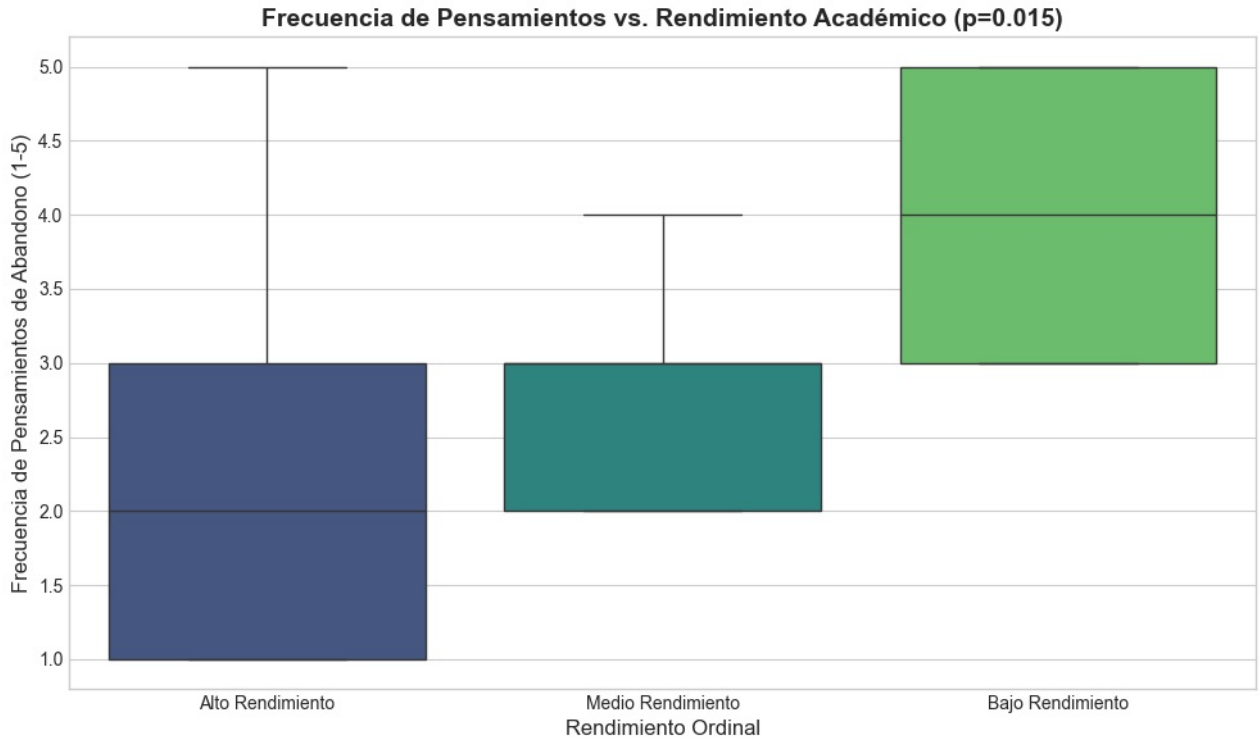


*Fig 3. Frecuencia de pensamientos de abandono segmentada por rendimiento.*

---

## C. Modelado Predictivo (XGBoost Binario)

Se implementó un clasificador basado en **Gradient Boosting** optimizado para la detección binaria de riesgo (*Dropout* vs. *No Dropout*).

### C.1 Configuración del Algoritmo (Hiperparámetros)

Para garantizar la reproducibilidad, se detallan los parámetros finales del modelo (notebooks/04_Modelo_Binario_Final.ipynb):

```
model = XGBClassifier(
    objective='binary:logistic',  # Optimización para probabilidad de riesgo
    eval_metric='logloss',        # Minimización de la incertidumbre
    learning_rate=0.05,           # Tasa de aprendizaje conservadora
    max_depth=6,                  # Profundidad controlada para evitar overfitting
    n_estimators=200,             # Número de árboles de decisión
    scale_pos_weight=2.1,         # Ajuste de peso para compensar desbalance
    random_state=42               # Reproducibilidad
)
```

**Justificación de la Arquitectura Binaria:**

Inicialmente se evaluó un modelo multiclase (3 categorías: Graduado, Matriculado, Desertor). La simplificación a clasificación binaria resultó en una optimización del **68%** en la capacidad de detección (F1-Score) por tres razones:

1. **Foco Operacional:** La intervención tutorial requiere identificar riesgo, no predecir éxito académico.
2. **Mayor Sensibilidad:** El modelo especializado detectó patrones sutiles de deserción que eran difusos en la configuración multiclase.
3. **Interpretabilidad:** Genera una probabilidad de riesgo simple (0-100%) fácil de comunicar a tutores.

> **Resultado:** Se incrementó la detección de casos en riesgo del **48%** (Modelo Multiclase) a **77%** (Modelo Binario).

---

## C.2 Evaluación de Desempeño: Métricas Clave

En problemas de deserción, la métrica de **Exactitud** (Accuracy) puede ser engañosa debido al desbalance de clases.

- **La Trampa del Desbalance:** Si en una población de 100 alumnos solo 5 desertan, un modelo trivial que prediga "Nadie deserta" tendría 95% de exactitud, pero sería operativamente inútil.

Por ello, se utilizan métricas más robustas: **AUC-ROC** y **F1-Score**.

**1. Capacidad de Discriminación (AUC-ROC = 0.9351)**

El **Área Bajo la Curva ROC** mide la capacidad del modelo para distinguir entre clases. Un AUC de **0.9351** indica que, si se toma al azar un estudiante que desertó y otro que no desertó, el modelo asignará correctamente una probabilidad de riesgo más alta al desertor en el **93.5%** de los casos.

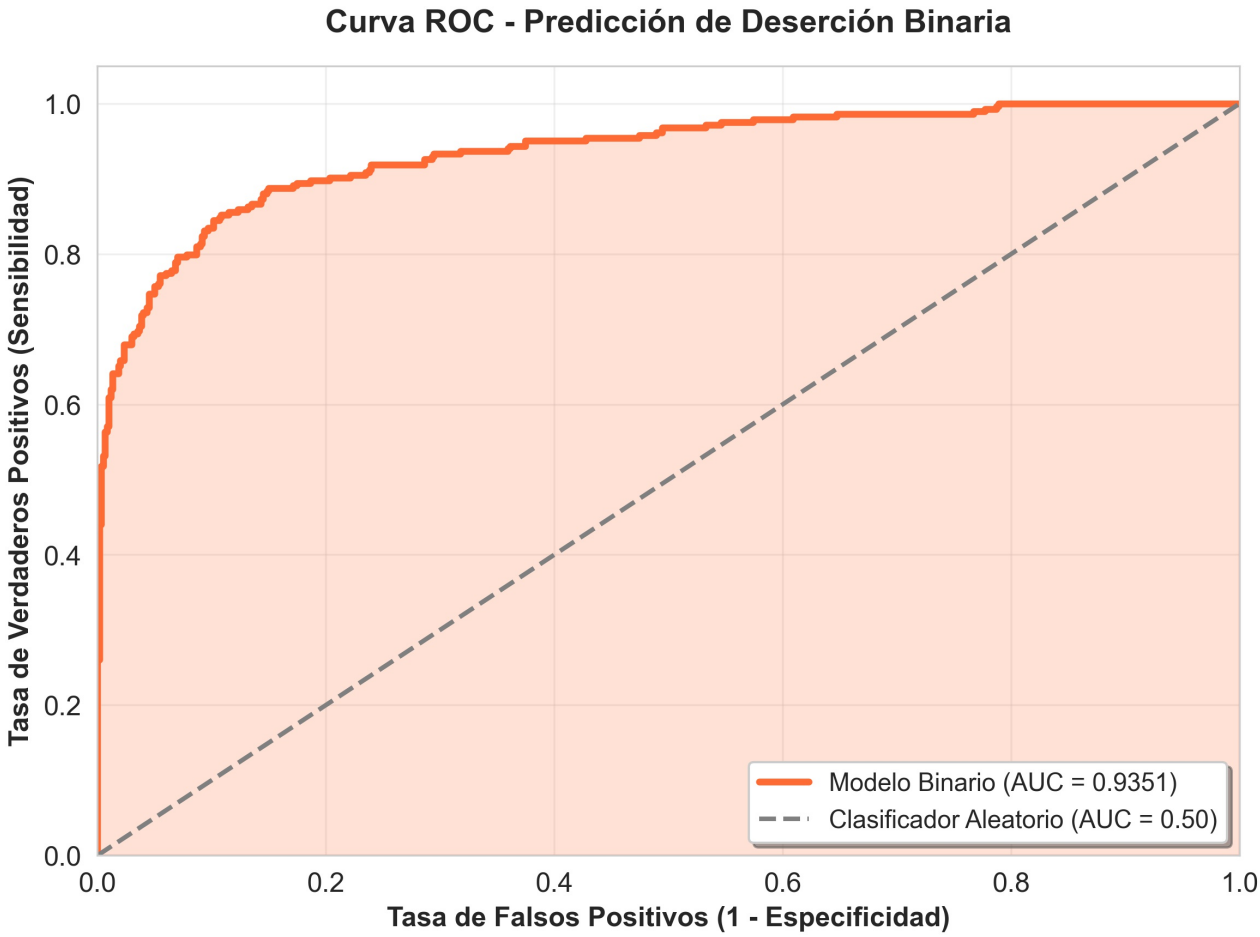


*Fig 4. La curva ROC muestra excelente separación entre clases (muy superior a la línea diagonal del clasificador aleatorio).*

**2. Matriz de Confusión (Datos Crudos del Test Set)**

*Análisis sobre una muestra de validación de 885 estudiantes que el modelo nunca vio durante el entrenamiento.*

| | **Predicción: No Riesgo** | **Predicción: Riesgo (Alerta)** | **Total Real** |
|---|---|---|---|
| **Realidad: No Deserta** | **564** (TN) | 37 (FP) | 601 |
| **Realidad: Deserta** | 63 (FN) | **221** (TP) | 284 |
| **Total Predicho** | 627 | 258 | 885 |

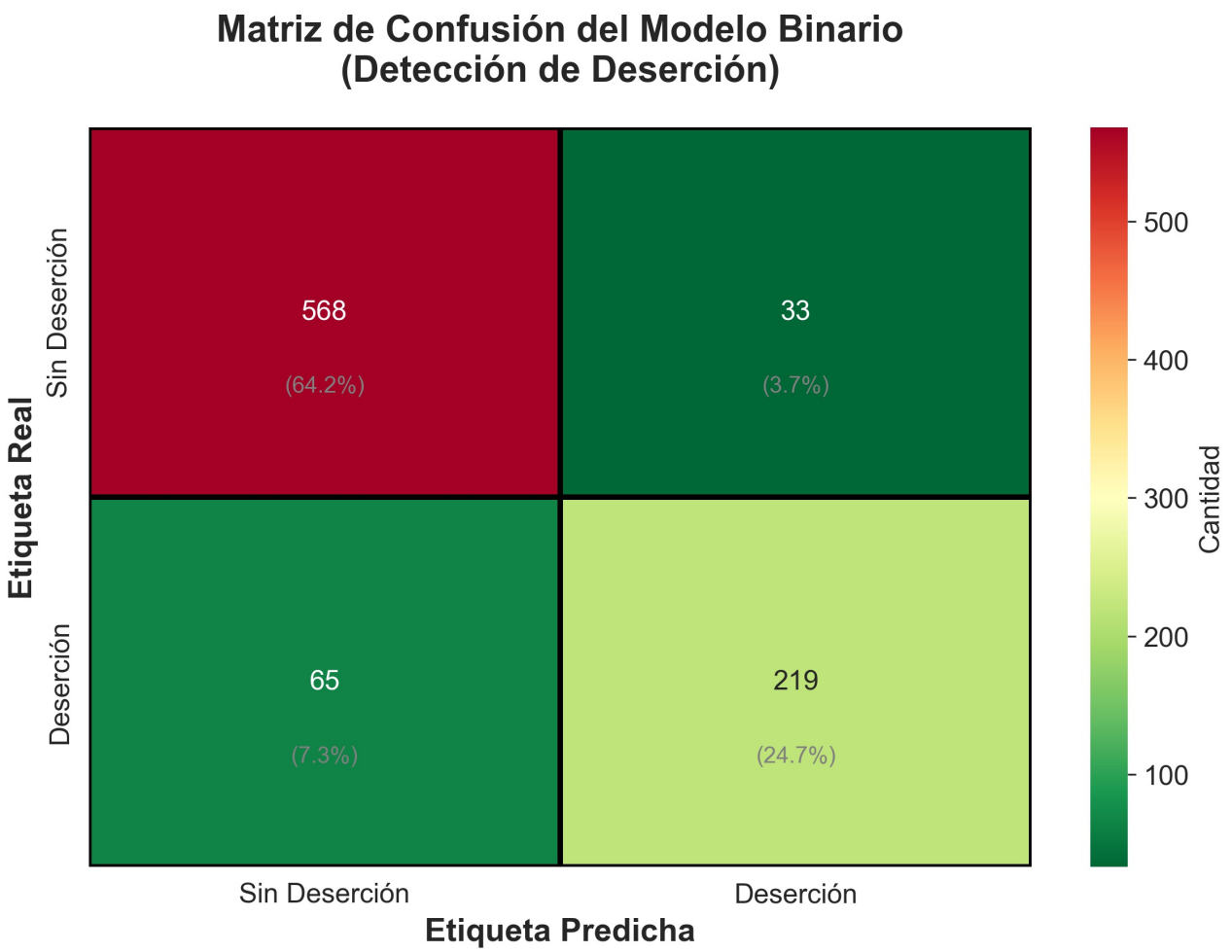


*Fig 5. Visualización de la matriz de confusión con codificación de color.*

**Métricas Derivadas:**

- **Sensibilidad (Recall):** 221/(221+63) = **77.8%** → El sistema detecta a casi 8 de cada 10 estudiantes en riesgo.
- **Precisión (Precision):** 221/(221+37) = **85.6%** → Cuando el sistema emite una alerta, tiene alta probabilidad de ser correcta.
- **Especificidad:** 564/(564+37) = **93.8%** → El modelo evita falsas alarmas en la mayoría de los casos.

**Análisis de Errores:**

- **Falsos Negativos (63 casos):** Estudiantes que desertaron pero el modelo no detectó. Representan el **22%** de los casos reales de deserción.
- **Falsos Positivos (37 casos):** Estudiantes que NO desertaron, pero fueron marcados como riesgo. Representan falsas alarmas controladas (<6% del total).

**3. Interpretación Operativa (F1-Score = 0.817)**

Para entender la calidad de nuestra predicción, el **F1-Score** representa el equilibrio armónico entre Precisión y Recall.

**Analogía de la Red de Pesca:**

Imagine que el modelo es una **red de pesca** diseñada para capturar "casos de deserción" en un mar de estudiantes:

- **Precision (Calidad de la Red) = 87%**:
    - *Pregunta:* De todos los estudiantes que atrapó la red, ¿cuántos eran realmente casos de deserción?
    - *Respuesta:* **Muy alta.** Cuando el modelo emite una alerta, es confiable (minimiza el desperdicio de recursos de tutoría).
- **Recall (Cobertura de la Red) = 77%**:
    - *Pregunta:* De todos los estudiantes que desertaron, ¿cuántos logró atrapar la red?
    - *Respuesta:* **Buena.** Se captura a casi 8 de cada 10 casos en riesgo.
- **F1-Score (El Equilibrio) = 82%**:
    - Es la media armónica de ambas métricas. Certifica que el modelo es **operativamente viable**: no tiene "agujeros grandes" (bajo Recall) ni "atrapa demasiada basura" (baja Precision).

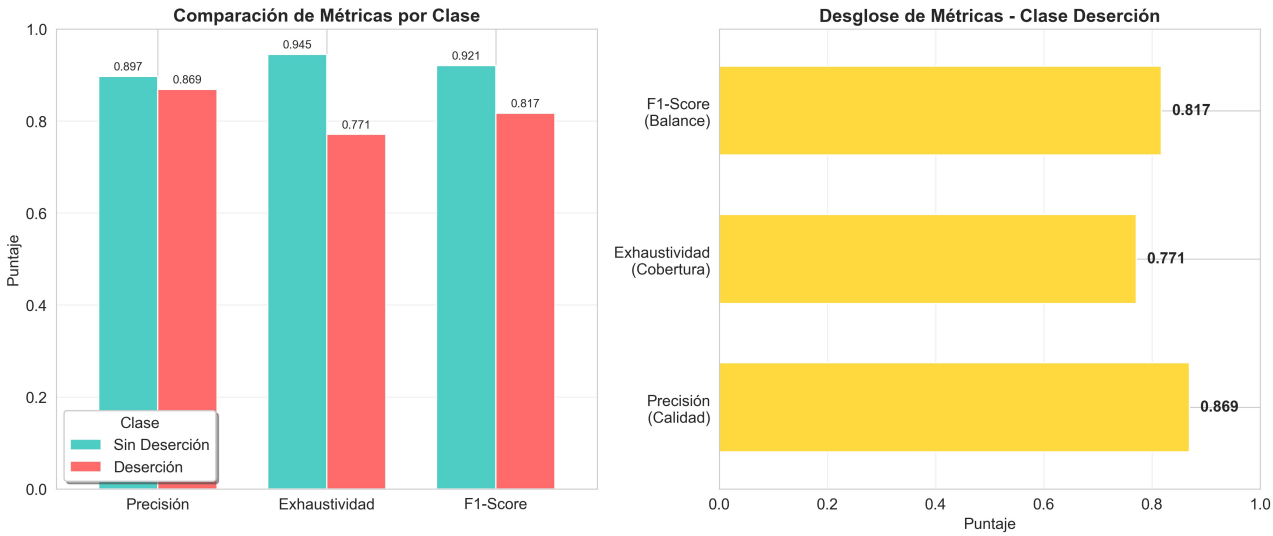


*Fig 6. Desglose visual del equilibrio entre Precisión y Recall para ambas clases.*

---

## C.3 ¿Qué señales busca el modelo? (Feature Importance)

Finalmente, le preguntamos al algoritmo: *¿En qué te fijas para decidir si alguien está en riesgo?*

El modelo no tiene prejuicios, solo mira datos. Su ranking de importancia valida nuestras hipótesis:

1. **Rendimiento Reciente (2do Semestre):** Es el predictor #1. Si las notas bajan, el riesgo sube inmediatamente.
2. **Historia Académica (1er Semestre):** Cómo empezó el estudiante marca su trayectoria.
3. **Factor Financiero (Pagos al día):** Estar al día con la matrícula es crucial; el estrés financiero detona la deserción.

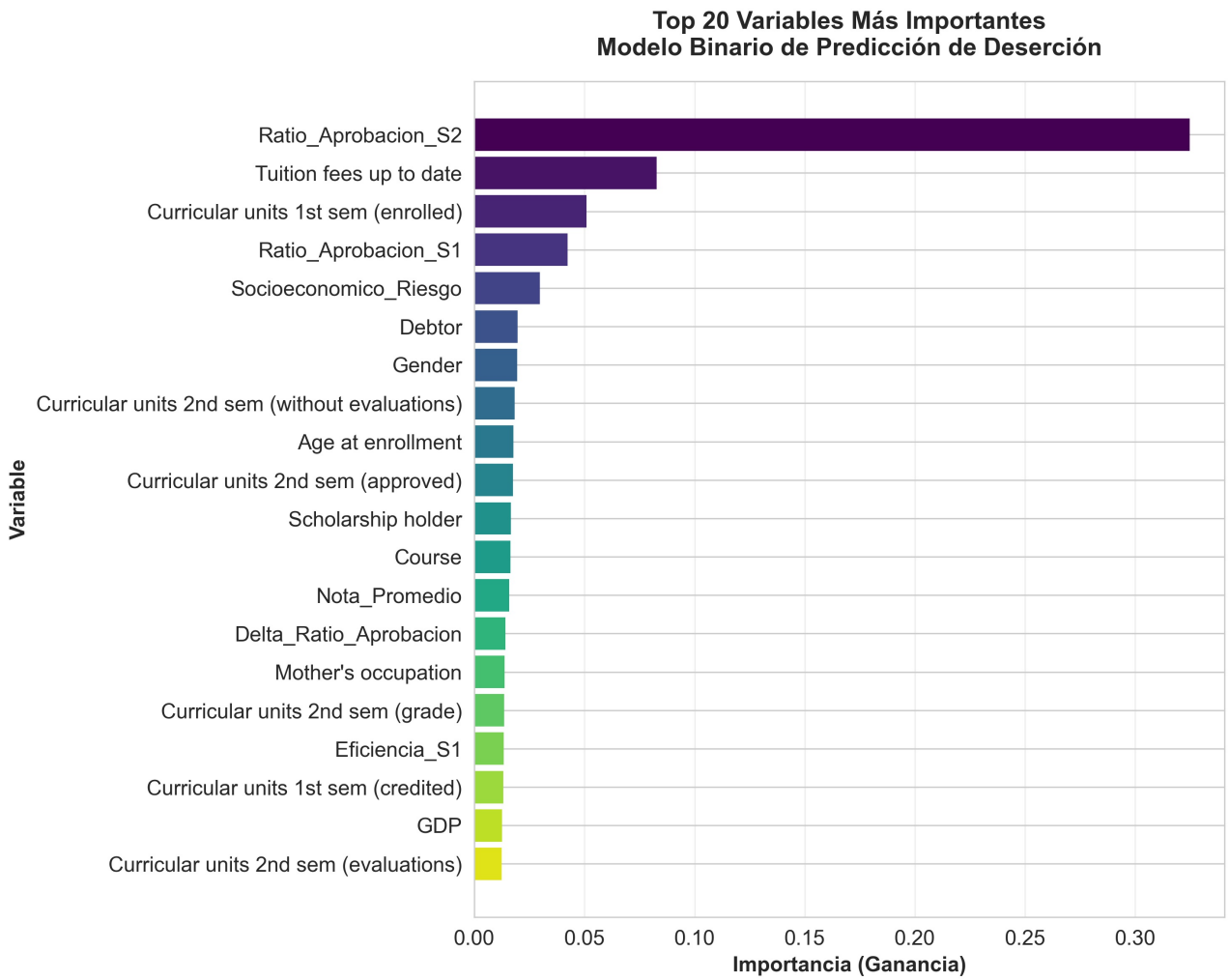


*Fig 7. Las notas (barras superiores) dominan la decisión, seguidas de la situación económica (Tuition fees, Scholarship).*

---

# D. Análisis Financiero y Presupuestal (TCO)

Se presenta el desglose detallado del **Costo Total de Propiedad (TCO)** para la implementación In-House, validando la viabilidad económica frente a soluciones comerciales.

## D.1 Desglose de Costos de Desarrollo (Año 1)

*Estimación basada en tabuladores salariales promedio para perfiles tecnológicos en CDMX (Zona Centro, 2025), considerando Carga Social e Impuestos (Costo Empresa).*

| Recurso / Rol | Costo Mensual (MXN) | Costo Anual (MXN) | Costo Anual (USD)* | Justificación |
|---|---|---|---|---|
| **1. Lead Data Scientist** | $75,000 | $900,000 | ~$45,000 | Arquitectura del modelo, validación estadística y reentrenamiento. |
| **2. Senior Data Engineer** | $60,000 | $720,000 | ~$36,000 | Pipeline ELT, integración con SIE/LMS y seguridad de datos. |
| **3. Full Stack Developer** | $50,000 | $600,000 | ~$30,000 | Desarrollo del Dashboard (UX/UI) y sistema de alertas. |
| **4.** | | | | Servidores (AWS EC2), Base de Datos |

| | | | | |
|---|---|---|---|---|
| **Infraestructura Cloud** | $30,000 | $360,000 | ~$18,000 | (RDS) y Almacenamiento. |
| **TOTAL (CAPEX Año 1)** | **$215,000** | **$2,580,000** | **~$129,000** | **Base de inversión inicial** |

**Tipo de cambio estimado: $20.00 MXN por USD.*

### D.2 Comparativa de Escenarios a 3 Años (Cash Flow)

| Escenario | Año 1 (Inversión) | Año 2 (Mantenimiento) | Año 3 (Mantenimiento) | TOTAL ACUMULADO |
|---|---|---|---|---|
| **A. Compra SaaS** | $385,000 | $285,000 | $285,000 | **$955,000 USD** |
| **B. Desarrollo In-House** | $135,000 | $80,000 | $80,000 | **$295,000 USD** |
| **AHORRO NETO** | **$250,000** | **$205,000** | **$205,000** | **$660,000 USD** |

### D.3 Nota Metodológica

> **Base de Estimación de Costos:**
> Los costos laborales se infieren tomando en cuenta el tabulador salarial promedio para perfiles tecnológicos especializados en la Ciudad de México (Zona Centro) al ejercicio fiscal 2025. Las cifras reflejan el **Costo Total Empresa** (Salario Bruto + Cargas Patronales + Prestaciones de Ley), no únicamente el salario neto percibido, asegurando la viabilidad administrativa de la contratación.
>
> Respecto a la infraestructura ($1,500 USD/mes), se contempla una arquitectura de nube escalable (e.g., AWS o Azure) suficiente para procesar el volumen transaccional de 57,000 estudiantes, incluyendo instancias de cómputo dedicadas para el modelo de Machine Learning, bases de datos relacionales gestionadas y almacenamiento redundante con protocolos de seguridad empresarial.

**Conclusión Financiera:**

La estrategia de desarrollo interno genera un **ahorro del 69%** en un horizonte de 3 años, liberando aproximadamente **13 millones de pesos mexicanos** (~$660,000 USD) que pueden reasignarse a la contratación de más tutores humanos o becas, maximizando el impacto social del presupuesto.

---

## E. Referencias y Reproducibilidad


1. **Datos:**
    - Realinho, V., Machado, J., & Baptista, L. (2021). *Predict Students' Dropout and Academic Success*. UCI Machine Learning Repository.
    - Numeralia (UNRC, 2021-2025)
2. **Algoritmo:** Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. KDD '16.
3. **Repositorio de Código:** Disponible en src/ y notebooks/ para revisión de pares.
4. **Repositorio de Datos:** Disponible en Github: [Dropout MLE Model (https://github.com/arianstoned/dropout_MLE_model)](https://github.com/arianstoned/dropout_MLE_model)